UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/870,613	05/31/2001	Scott J. Broussard	AUS920010261US1	1790

35617 7590 08/25/2005

DAFFER MCDANEIL LLP
P.O. BOX 684908
AUSTIN, TX 78768

EXAMINER	
BONSHOCK, DENNIS G	

ART UNIT	PAPER NUMBER
2173	

DATE MAILED: 08/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/870,613
Filing Date: May 31, 2001
Appellant(s): BROUSSARD, SCOTT J.

Kevin L. Daffer (reg. 34,146)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 4-07-2005.

(1) Real Party in Interest

A statement identifying the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

A statement identifying the related appeals and interferences which will directly affect or be directly affected by or have a bearing on the decision in the pending appeal is contained in the brief.

(3) Status of Claims

The statement of the status of the claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Invention

The summary of invention contained in the brief is correct.

(6) Issues

The appellant's statement of the issues in the brief is correct.

(7) Grouping of Claims

Appellant's brief includes a statement that claims 1-4, 8, 10, 11, and 17-20 do not stand or fall together and provides reasons as set forth in 37 CFR 1.192(c)(7) and (c)(8).

Appellant's brief includes a statement that claims 12 and 14 do not stand or fall together and provides reasons as set forth in 37 CFR 1.192(c)(7) and (c)(8).

(8) Claims Appealed

The copy of the appealed claims contained in the Appendix to the brief is correct.

(9) Prior Art of Record

6,727,918	Nason	4-2004
5,327,529	Fults et al.	7-1994

(10) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-4, 8, 10-12, 14, and 17-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Nason, Patent #6,727,918 and Fults et al., Patent #5,327,529, hereinafter Fults.

With regard to claim 1, Nason teaches a system for providing a second graphical interface with a primary graphical interface (see column 5, lines 18-22), a display (see column 5, lines 33-44), a memory (see column 3, line 65 though column 4, line 3), platform independent components for implementing the system (see column 5, lines 45-63), and implementing the system using content and operating software in JAVA (see column 5, lines 60-63). JAVA is known in the art to comprise GUIs such as AWT, a JAVA GUI that is known to rely on the native GUI of the computer on which the

application runs, and SWING, a JAVA GUI that runs uniformly on any native platform (see Microsoft Computer Dictionary pages 13 and 505). Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components where the elements are specifically list oriented. Fults teaches, a system of using two different user interfaces in the same application (see column 24, line 38 through column 25, line 10), similar to that of Nason, but further teaches a Generic User Interface Object Library and Controller and Specific User Interface Interpreters that map I/O to the Specific User interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21), and further teaches the implementation of the system using lists (see column 5, lines 35-51). It would have been obvious to one of ordinary skill in the art, having the teachings of Nason and Fults before him at the time the invention was made to modify the system of jointly using two interfaces of Nason to include a peer component that routes the invocations between elements of Fults. One would have been motivated to make such a combination because this allows using an interface other that the one that the system expects.

With regard to claim 2, which teaches an application program that creates the list data prior to fetching the list from memory, Nason teaches, in column 5, lines 45-51, allocating space for content before creating it.

With regard to claim 3, which teaches the list only created once in memory, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA,

which is known the use the SWING API, which is disclosed in the specification on page 31 to obviate the need for a redundant memory array.

With regard to claim 4, which teaches an image upon the display being created independent of the operating system, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA, which is known the use the SWING API, which is a display API that doesn't rely on any native platform.

With regard to claim 8, which teaches the system of platform independent software components comprising Java swing API, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA, which is known the use the SWING API (see Microsoft Computer Dictionary pages 13 and 505).

With regard to claim 10, which teaches the platform independent software component is a choice or list control, Fults further teaches, in column 5, lines 35-51, the implementation of the system using lists (see column 5, lines 35-51).

With regard to claim 11, which teaches the application program being in java programming language, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA.

With regard to claim 12, Nason teaches a system for providing a second graphical interface with a primary graphical interface (see column 5, lines 18-22), a display (see column 5, lines 33-44), a memory (see column 3, line 65 though column 4, line 3), platform independent components for implementing the system (see column 5, lines 45-63), and implementing the system using content and operating software in JAVA (see column 5, lines 60-63). JAVA is known in the art to comprise GUIs such as

AWT, a JAVA GUI that is known to rely on the native GUI of the computer on which the application runs, and SWING, a JAVA GUI that runs uniformly on any native platform (see Microsoft Computer Dictionary pages 13 and 505). Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components where the elements are specifically list oriented. Fults teaches, a system of using two different user interfaces in the same application (see column 24, line 38 through column 25, line 10), similar to that of Nason, but further teaches a Generic User Interface Object Library and Controller and Specific User Interface Interpreters that map I/O to the Specific User interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21), and further teaches the implementation of the system using lists (see column 5, lines 35-51). It would have been obvious to one of ordinary skill in the art, having the teachings of Nason and Fults before him at the time the invention was made to modify the system of jointly using two interfaces of Nason to include a peer component that routes the invocations between elements of Fults. One would have been motivated to make such a combination because this allows using an interface other that the one that the system expects.

With regard to claim 14, which teaches the first platform independent software component being a peer component emulating a platform dependent peer component, Fults further teaches a component that maps the User Interface to a new User Interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21)

With regard to claim 17, Nason teaches a system for providing a second graphical interface with a primary graphical interface (see column 5, lines 18-22), a display (see column 5, lines 33-44), a memory (see column 3, line 65 though column 4, line 3), platform independent components for implementing the system (see column 5, lines 45-63), and implementing the system using content and operating software in JAVA (see column 5, lines 60-63). JAVA is known in the art to comprise GUIs such as AWT, a JAVA GUI that is known to rely on the native GUI of the computer on which the application runs, and SWING, a JAVA GUI that runs uniformly on any native platform (see Microsoft Computer Dictionary pages 13 and 505). Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components where the elements are specifically list oriented. Fults teaches, a system of using two different user interfaces in the same application (see column 24, line 38 through column 25, line 10), similar to that of Nason, but further teaches a Generic User Interface Object Library and Controller and Specific User Interface Interpreters that map I/O to the Specific User interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21), and further teaches the implementation of the system using lists (see column 5, lines 35-51). It would have been obvious to one of ordinary skill in the art, having the teachings of Nason and Fults before him at the time the invention was made to modify the system of jointly using two interfaces of Nason to include a peer component that routes the invocations between

elements of Fults. One would have been motivated to make such a combination because this allows using an interface other that the one that the system expects.

With regard to claim 18, which teaches the platform independent application program being written in JAVA, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA.

With regard to claim 19, which teaches the second software component being a JAVA swing component, Nason teaches, in column 5, lines 60-63, the system being implemented with JAVA, which is known the use the SWING API (see Microsoft Computer Dictionary pages 13 and 505).

With regard to claim 20, which teaches the first software component being a peer component, which servers as an interface between the platform dependent invocations of the application program written in Java an the Swing software components, Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components, or the elements being specifically list oriented. Fults further teaches a component that maps the User Interface to a new User Interface, which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21).

This rejection is set forth in a prior Office Action, mailed on 11-16-2004.

(11) Response to Argument

Group I:

With respect to the group of claims including Claims 1-4, 8, 10, 11, and 17-20, the Appellant's arguments are focused on the limitations regarding the existence of a platform independent software component, a platform independent application program, and a platform independent peer component in the cited references. More specifically, as stated from representative Claim 1, the limitation argued is:

> *"(i) a platform-independent software component for fetching list data from memory and for producing a display image of the list data without invoking platform-dependent display routines, and (ii) a platform-independent peer component for intercepting invocations of platform-dependent display routines from a platform-independent application program and for routing the intercepted invocations to the platform-independent software component."*

Since the interpretation of the limitation is the basis for the arguments, the Examiner's interpretation is now given. The claim, as interpreted by the examiner, pertains to the existence of an element (peer component) that intercepts a platform dependent display routine (from the application program), and directs it to a platform independent software component for display. As stated in the eighth paragraph of MPEP 2101[R2].II.C.,

"*Office personnel are to give claims their broadest reasonable interpretation in light of the supporting disclosure*. In re Morris, 127 F.3d 1048, 1054-55, 44 USPQ2d 1023,1027-28 (Fed. Cir. 1997)."

Based on the interpretation of the claim limitations being argued, the Examiner will now explain how the teachings of the Nason and Fults et al., hereinafter Fults, references are within the scope of these limitations.

Nason teaches a system for providing a second graphical interface with a primary graphical interface (see column 5, lines 18-22), a display (see column 5, lines 33-44), a memory (see column 3, line 65 though column 4, line 3), platform independent components for implementing the system (see column 5, lines 45-63), and implementing the system using content and operating software in JAVA (see column 5, lines 60-63). JAVA is known in the art to comprise GUIs such as AWT, a JAVA GUI that is known to rely on the native GUI of the computer on which the application runs, and SWING, a JAVA GUI that runs uniformly on any native platform (see Microsoft Computer Dictionary pages 13 and 505). Nason teaches an Alternate Display Content Controller (ADCC) (peer component) that contains and Application Program Interface (API) that intercepts all calls to a graphics device interfaces (GDI) (native to the operating systems) (platform dependent display routines) and routes them to a complementary GUI display driver [see column 20, line 46 through column 21, line 28]. This complementary

GUI display is capable of being independent of the native operating system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components with the elements being specifically list oriented. Fults teaches, a system of using two different user interfaces in the same application (see column 24, line 38 through column 25, line 10), similar to that of Nason, but further teaches a Generic User Interface Object Library and Controller and Specific User Interface Interpreters that map I/O to the Specific User interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21), and further teaches the implementation of the system using lists (see column 5, lines 35-51).

The examiner will now address the individual arguments and statements made by Appellant.

From pages 8 of the Appeal Brief, from the second paragraph, the Appellant argues that "Nason fails to provide teaching or suggestion for the platform-independent software component".

The examiner respectfully contends that Nason does show a complementary GUI display driver (see column 20, line 46 through column 21, line 28) and that this complementary GUI display is capable of being independent of the native operating

system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). The applications are shown to be combined with the kernel which is operating system independent.

From page 8 of the Appeal Brief, from the third and forth paragraph, the Appellant argues that "Fults cannot be combined with Nason to provide teaching or suggestion the presently claimed peer or software components."

The examiner respectfully contends that Fults supplements the teaching of a peer component that intercepts all calls to a graphics device interfaces (GDI) (native to the operating systems) (platform dependent display routines) and routes them to a complementary GUI display driver (see column 20, line 46 through column 21, line 28), of Nason, by providing for a peer component (GUIOLC and SUII) that maps input/output (such as an output to a display) requirements of an Application to the SUITC (specific user interface) under which the display is provided (see column 24, lines 38-67). Examiner would like to further point out the a software component, as claimed, can be any group of instructions in a computer system (including a application, a program, a operating system, etc.).

From page 9 of the Appeal Brief, from the second and third paragraph, the Appellant argues that "Fults discloses a "generic" or platform-independent user interface, which when run by a particular operating system, invokes a platform-

independent display routine form the operating system to produce images of the gadgets", which the Appellant compares to AWT (a platform dependent display routine.

The examiner respectfully contends that the Appellant is misinterpreting the combination made by the Examiner in the rejection under 35 U.S.C. 103, of record. Fults supplements the teachings of Nason, which is relied upon for the teaching of platform-independent display of platform dependent routines. Fults is merely relied upon for a clearer recitation of a peer component linking and application program and an software component capable of receiving a display request and implementing it (see column 24, lines 59-67).

From page 10 of the Appeal Brief, from the fourth paragraph, the Appellant argues that "The Examiner has failed to adequately support and/or establish a prima facie ground of obviousness.

In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, both references teach the use of a peer component to route the display of a particular routine to another software element. Fults further supports this relationship by specifying the origin of the

display data being from an application (see column 24, lines 59-67) and the specific routine being of list data (see column 8, lines 36-55). This further specifies a use for Nason, in the displaying of list data, in a manner similar to Nason (through a secondary user interface).

Group II:

With respect to the group of claims including Claims 12 and 14, the Appellant's arguments are focused on the limitations regarding the existence of a platform independent software component, a platform independent application program, and an intercepting platform independent (peer) component in the cited references. More specifically, as stated from representative Claim 12, the limitation argued is:

> *"Nason and Fults each fail to disclose a method for generating a display image, where the method includes intercepting platform-dependent invocations by a first platform-independent software component, and generating a display image using a second platform-independent software component without creating a copy of list data stored by an application program"*

Since the interpretation of the limitation is the basis for the arguments, the Examiner's interpretation is now given. The Examiner asserts the specification of the present application was a specification that was made to fit at least 8 other cases, and wasn't particularly directed to this present claim making interpretation of the claim primarily based on the word for word meaning in the art. The claim, as interpreted by

the examiner, pertains to the existence of an software element (peer component) that intercepts a platform dependent display routine (from the application program), and directs it to a second platform independent software component for display. As stated in the eighth paragraph of MPEP 2101[R2].II.C.,

> "*Office personnel are to give claims their broadest reasonable interpretation in light of the supporting disclosure.* In re Morris, 127 F.3d 1048, 1054-55, 44 USPQ2d 1023,1027-28 (Fed. Cir. 1997)."

Based on the interpretation of the claim limitations being argued, the Examiner will now explain how the teachings of the Nason and Fults et al., hereinafter Fults, references are within the scope of these limitations.

Nason teaches a system for providing a second graphical interface with a primary graphical interface (see column 5, lines 18-22), a display (see column 5, lines 33-44), a memory (see column 3, line 65 though column 4, line 3), platform independent components for implementing the system (see column 5, lines 45-63), and implementing the system using content and operating software in JAVA (see column 5, lines 60-63). JAVA is known in the art to comprise GUIs such as AWT, a JAVA GUI that is known to rely on the native GUI of the computer on which the application runs, and SWING, a JAVA GUI that runs uniformly on any native platform (see Microsoft Computer Dictionary pages 13 and 505). Nason

teaches an Alternate Display Content Controller (ADCC) (peer component) that contains and Application Program Interface (API) that intercepts all calls to a graphics device interfaces (GDI) (native to the operating systems) (platform dependent display routines) and routes them to a complementary GUI display driver [see column 20, line 46 through column 21, line 28]. This complementary GUI display is capable of being independent of the native operating system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). Nason teaches a peer component for dealing with two different interfaces, however doesn't specifically disclose a peer component that routes the invocations between software components with the elements being specifically list oriented. Fults teaches, a system of using two different user interfaces in the same application (see column 24, line 38 through column 25, line 10), similar to that of Nason, but further teaches a Generic User Interface Object Library and Controller and Specific User Interface Interpreters that map I/O to the Specific User interface which the application is to be presented to the user in (see column 24, line 38 through column 25, line 10 and figure 21), and further teaches the implementation of the system using lists (see column 5, lines 35-51).

The examiner will now address the individual arguments and statements made by Appellant.

From page 12 of the Appeal Brief, from the first paragraph, the Appellant argues that "the cited art fails to teach, suggest or provide motivation for a platform-independent software component" which "draws from a library of commands that are independent of an operating system for producing the display image of the list data".

The examiner respectfully contends that Nason does show a complementary GUI display driver (see column 20, line 46 through column 21, line 28) and that this complementary GUI display is capable of being independent of the native operating system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). The applications are further shown to be combined with the kernel, which is operating system independent.

From page 12 of the Appeal Brief, from the third paragraph, the Appellant argues that the cited art fails to "provide teaching or suggestion for a platform-independent software component that generates a display list image without creating a copy of list data stored by a platform-independent application program."

The examiner respectfully contends that Nason does show a complementary GUI display driver (see column 20, line 46 through column 21, line 28) and that this complementary GUI display is capable of being independent of the native operating system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). This argument goes along with the Applicants assertion that the User Interface being used

for display is a AWT API. This is clearly not true given that the software component being used for display is clearly stated to "operate independent of the native operating system", and further "remain executing, even when the operating system fails." AWT would clearly fail in this case given that AWT works in connection with the native GUI for display. The applications are further shown to be combined with the kernel, which is operating system independent.

From pages 12 and 13 of the Appeal Brief, from the fourth paragraph of page 12 to the second paragraph of page 13, the Appellant argues that the system of Nason is being implemented through the use of AWT (abstract windowing toolkit), and that this would not provide a platform independent software component for intercepting platform-dependent invocations, and a second platform-independent software component for generating a display list image without creating a copy of list data stored by a platform-independent application program (as in AWT).

The examiner respectfully contends that Nason teaches an Alternate Display Content Controller (ADCC) (peer component) that contains and Application Program Interface (API) that intercepts all calls to graphics device interfaces (GDI) (native to the operating systems) (platform dependent display routines) and routes them to a complementary GUI display driver [see column 20, line 46 through column 21, line 28]. This complementary GUI display is capable of being independent of the native operating system (see column 4, line 61 through column 5, line 4 and column 5, lines 45-63). Nason further states that this user interface used for display is from a software component that operates

independent of the native operating system, and further would "remain executing, even when the operating system fails", and provides no mention to redundant memory storage. This argument goes along with the Applicants assertion that the User Interface being used for display is a AWT API. This is clearly not true given that the software component being used for display is clearly stated to "operate independent of the native operating system", and further "remain executing, even when the operating system fails." AWT would clearly fail in this case given that AWT works in connection with the native GUI for display.

From page 13 of the Appeal Brief, from the third paragraph, the Appellant argues that "There is no motivation to modify the teachings of Nason or Fults to provide the presently claimed method for generating a display image, where the method includes intercepting platform-dependent invocations by a first platform-independent software component, and generating a display image using a second platform-independent software component without creating a copy of list data stored by an application program."

In response to applicant's argument that the examiner's conclusion of obviousness is based upon improper hindsight reasoning, it must be recognized that any judgment on obviousness is in a sense necessarily a reconstruction based upon hindsight reasoning. But so long as it takes into account only knowledge which was within the level of ordinary skill at the time the claimed invention was made, and does not include knowledge gleaned only from the applicant's disclosure, such a

reconstruction is proper. See *In re McLaughlin*, 443 F.2d 1392, 170 USPQ 209 (CCPA 1971).

Both references teach the use of a peer component to route the display of a particular routine to another software element. Fults just further specifies the origin of the display data being from an application (see column 24, lines 59-67) and the specific routine being of list data (see column 8, lines 36-55). This further specifies a use for Nason, in the displaying of list data, in a manner similar to Nason (through a secondary user interface).

From page 14 of the Appeal Brief, from the fourth paragraph, the Appellant argues that "The Examiner has failed to adequately support and/or establish a prima facie ground of obviousness.

In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, both references teach the use of a peer component to route the display of a particular routine to another software element. Fults just further specifies the origin of the display data being from an application (see column 24, lines 59-67) and the specific routine being of list data (see

column 8, lines 36-55). This further specifies a use for Nason, in the displaying of list data, in a manner similar to Nason (through a secondary user interface).

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Conferees:

Dennis G. Bonshock
August 15, 2005

John W. Cabeca
Supervisory Patent Examiner
August 15, 2005

JOHN CABECA
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER

KRISTINE KINCAID
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

DAFFER MCDANEIL LLP
P.O. BOX 684908
AUSTIN, TX 78768